JPAZ2 framework

JPAZ2 je framework pre podporu výučby princípov programovania a základných koncepcií objektovo-orientovaného programovania v **Jave pre začiatočníkov**. Framework je postavený na využití korytnačej grafiky a *prieskumníka objektov*. Prieskumník objektov umožňuje zobraziť a meniť hodnoty vlastností objektov a tiež spúšťať metódy objektov.

Ďalším významným cieľom frameworku je byť nástrojom, ktorý umožní záujemcom o programovanie tvoriť vlastné kreatívne miniaplikácie a programy (napr. 2D hry) bez nutnosti "zápasiť" s technickými detailami Javy. Tieto technické detaily sú v JPAZ2 ukryté do starostlivo navrhnutého systému tried. Prah na programovanie jednoduchých interaktívnych a vizuálnych aplikácii v Jave sa použitím JPAZ2 posúva výrazne nižšie - nie je treba riešiť vlákna a synchronizáciu vlákien, animovanie objektov, použitie komplexných tried Java API pre prácu so zvukom a grafikou, ...

Projekt JPAZ2 na GitHub-e

- JPAZ2 na GitHub-e - vždy najnovšie verzie JPAZu

JPAZ2 a Maven

- JPAZ2 v Maven repozitároch

Dokumentácia

- JavaDoc dokumentácia

Demonštračné mikroprojekty

- **SnowflakesCatcher** (SnowflakesCatcher.jar, Zdrojové kódy na GitHube)
  - komplexná a komentovaná ukážka rozšírení JPAZ2
  - verzia `SnowflakesCatcher2` obsahuje podporné triedy, ktoré zatiaľ nie sú súčasťou JPAZu.
- RaceTrack (RaceTrack.jar, RaceTrack.zip)
  - ukážka spracovávania stlačenia klávesov
  - ukážka využitia časovača
  - ukážka toho, ako pridať obrázky do projektu tak, aby sa exportovali spolu so spustiteľným jar-kom
- ClickMover (ClickMover.jar, ClickMover.zip)
  - ukážka animovaného presúvania objektov odštartovaného udalosťou myši

Kľúčové triedy

- `ObjectInspector` - základná trieda schopná "skúmať" iné objekty v run-time (počas behu programu) - zobrazovať ich vlastnosti a spúšťať definované metódy
- `Turtle` - základná trieda reprezentujúca "korytnačky" v korytnačej 2D grafike a zároveň grafický objekt v ploche (`Pane`)
  - kresliace schopnosti korytnačej grafiky
  - otlačenie svojho tvaru do plochy (`Pane`) - umožňuje kresliť obrázky zo súboru do plochy
  - môže mať tvar definovaný statickým alebo animovaným obrázkom (komplexné tvary sú budované triedou `ImageShape.Builder`) - aj s možnosťou definovania rýchlosti animácie
  - korytnačka je vždy umiestnená v nejakej ploche (`Pane`)
- `Pane` - základná trieda reprezentujúca kresliacu plochu (metafora: papier so zadanými rozmermi)

    - môže obsahovať korytnačky a **iné plochy** (`Pane`-s)
    - možnosť definovať stred rotácie, smer natočenia a polohu stredu rotácie
    - nastaviteľná transparentnosť
    - môže reagovať na udalosti (klávesnice aj myši) - Upozornenie: Korytnačky nevedia reagovať na udalosti. Ak je to v programe potrebné, musí sa o to postarať plocha, v ktorej sa nachádza "korytnačka" (grafický objekt)
    - nastaviteľné tikanie (perióda s akou je volaná metóda `onTick`)
- `ImageShape` - reprezentuje tvar korytnačky (grafického objektu), ktorý je určený nejakým obrázkom
    - táto trieda je náhradou `ImageTurtleShape`
- `ImageShape.Builder` - reprezentuje objekt, pomocou ktorého je možné bližšie špecifikovať tvar korytnačky z grafického súboru
    - umožňuje nastaviť stred rotácie, počet náhľadov (views) a počet snímkov (frames) tvaru
    - náhľad (view) reprezentuje pohľad na tvar - napr. v závislosti od stavu grafického objektu, či jeho natočenia
    - snímka (frame) reprezentuje jeden snímok animácie - v animovaných tvaroch musia mať všetky náhľady (views) rovnaký počet snímkov
    - umožňuje nastaviť transparentnosť tvaru
    - umožňuje načítať animované tvary z neanimovaného formátu (t.j. nielen z animované gif)
- `AudioClip` - reprezentuje zvukový súbor (nahrávku) a poskytuje jednotné rozhranie pre jeho prehrávanie nezávisle od formátu zvukového súboru
    - podporované formáty: mid (syntetizovaná hudba), aiff, au a wav (nekomprimované audio)
    - pri importe pozor na zámenu s inými triedami s názvom `AudioClip` v Jave (v iných balíkoch)
    - narozdiel od iných tried v Jave (napr. `Clip`) tu nie je obmedzenie na veľkosť zvukového súboru a počet súčasne prehrávaných zvukov
    - pre zvukové efekty častých akcií v aplikácii odporúčame prednačítanie súboru do pamäte (parameter konštruktora) a prehrávanie cez `playAsActionSound` (umožňuje súčasne viac krát prehrávať ten istý zvuk - napr. viac "výbuchov" v hre naraz)
- `JPAZPanel` - základná trieda, ktorá prepája kresliace plochy (`Pane`-s) a Swing aplikácie.
    - zobrazuje jednu kresliacu plochu (`Pane`)
    - umožňuje aktivovať zarovnanie (`align`) plochy (`Pane`) a panelu (`JPAZPanel`) - rozmery plochy sa prispôsobujú rozmerom panelu a naopak
    - umožňuje zmeniť zobrazenú kresliacu plochu (`Pane`) - okamžite alebo s vizuálnym efektom prechodu (`TransitionEffect`, podobne ako efekty prechodov snímok v PowerPoint-e)
- `JPAZWindow` - ako `JPAZPanel`, ale kresliaca plocha sa zobrazuje v okne
    - použitie `JPAZWindow` je preferované pred rozširovaním triedy `WinPane` (v kóde konštruktora triedy rozširujúcej `WinPane` nie je garantované, že `onXYZ` metódy budú volané až po skončení tohto konštruktora).

Poznámky k synchronizácii

- JPAZ2 je navrhnutý tak, aby všetky `onXYZ` všetkých tried JPAZu boli vykonávané v EDT vlákne Swing-u. Vďaka tomu nie je potrebná žiadna synchronizácia (pokiaľ nevytvárate vlastné vlákna). Pokiaľ nemáte s vláknami a ich synchronizáciou dostatočné skúsenosti, odporúčame vlastné vlákna nevytvárať.
- Pokiaľ nie je uvedené inak, metódy objektov tried JPAZu sú implementované ako `thread-safe` (výnimkou je napríklad aj `ImageShape.Builder`).
- Kód metódy `main` je vykonávaný v inom vlákne, než všetky `onXYZ` metódy. Odporúčame preto v tejto metóde zrealizovať len vytvorenie objektov aplikácie. Pozor tiež na verejné statické referenčné

premenné inicializované v metóde `main` (môžu byť ešte neinicializované, t.j. `null`, keďže ich inicializácia neprebehla).

- Pre skúsených programátorov: Pokiaľ sa potrebujete synchronizovať voči iným akciám JPAZu, použite zámok `JPAZUtilities.getJPAZLock()` - pozor, z kódu držiaceho tento zámok nezískavajte iné zámky (obzvlášť sa nesynchronizujte s EDT vláknom Swing-u, napr. cez `SwingUtilities.invokeAndWait`) - môže to mať za následok deadlock.

FAQ

V projekte chcem umožniť používateľovi pri ukladaní stavu aplikácie (napr. hry) zvoliť si názov súboru. Ako sa dá naprogramovať vlastný "Uložiť ako" dialóg?

Na tento účel je možné využiť triedu JFileChooser. Akurát v prípade JPAZ projektov (JPAZ2 kvôli zjednodušeniu skrýva Swing/AWT komponenty, ktoré interne používa) môže problém určiť tzv. rodičovský komponent ako parameter metód `showOpenDialog` resp. `showSaveDialog` (okrem iného rodičovský komponent určuje, kde sa dialógové okno zobrazí, resp. voči čomu sa vycentruje). Tieto metódy však akceptujú aj parameter `null` (dialógové okno bude v strede obrazovky). Kvôli synchronizácii s EDT Swingu by vytváranie dialógového okna malo byť realizované len v rámci vykonávania onMouseXYZ, resp. onKeyXYZ metód.

```
protected void onMousePressed(int x, int y, MouseEvent detail) {
        JFileChooser fc = new JFileChooser();
        int volba = fc.showOpenDialog(null);
        if (volba == JFileChooser.APPROVE_OPTION) {
                System.out.println("Zvolený súbor: "
                                        + fc.getSelectedFile().getName());
        } else {
                System.out.println("Žiaden súbor nebol zvolený");
        }
}
```

Viac informácií: http://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html

V projekte chcem po kliknutí zobraziť nejaké jednoduché okno s hláškou alebo požiadať používateľa, aby nejako zadal nejaké údaje. Dá sa to?

Na jednoduché "hlášky" alebo zadávanie hodnôt je užitočným pomocníkom trieda JOptionPane, ktorá poskytuje mnoho užitočných statických metód. Tak ako pri predošlej otázke (dialógové okná na uloženie súboru) platí, že ako rodičovský komponent je treba uviesť `null` a tieto metódy je treba volať len z `onMouseXYZ`, resp. `onKeyXYZ` metód. Okrem týchto metód je s EDT Swingu synchronizovaná aj metóda `onTick` objektov triedy `TickTimer` (ak v rámci parametrov konštruktora nebolo explicitne nastavené, že vytváraný objekt triedy `TickTimer` nemá byť synchronizovaný s EDT Swingu).

```
protected void onMousePressed(int x, int y, MouseEvent detail) {
        String zadanaHodnota = JOptionPane.showInputDialog("Zadaj hodnotu");
        System.out.println(zadanaHodnota);

        JOptionPane.showMessageDialog(null, "Hláška: " + zadanaHodnota);
}
```

Poznámka: Aj keď platí, že metódy `onXYZ` by mali zbehnúť čo najrýchlejšie (žiadne dlhotrvajúce výpočty alebo `JPAZUtilities.delay`), v prípade otvárania modálnych dialógových okien s

čakaním za používateľským vstupom nie je problém.